A PRELIMENERY REPORT ON

AUTOMATED RESUME SCREENING BOT

SUBMITTED TO THE VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY,

PUNE

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF TECHNOLOGY (COMPUTER ENGINEERING)

SUBMITTED BY

Darshan Patil – 22210104

Shruti Dawange – 22320170

DEPARTMENT OF COMPUTER ENGINEERING

BRACT’S

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY

SURVEY NO. 3/4, KONDHWA (BUDRUK), PUNE – 411048, MAHARASHTRA (INDIA).

01. Introduction

1.1 Overview

The Automated Resume Screening Bot is a robotic process automation (RPA) solution developed using UiPath to simplify and speed up the process of matching candidate resumes with job descriptions. In modern recruitment, large volumes of resumes are received for each job posting. This makes manual screening inefficient, inconsistent, and time-consuming. The bot automates the task of reading resumes, comparing them with job descriptions, identifying matching skills, calculating match scores, and generating an Excel report that helps recruiters make quick and data-driven hiring decisions.

1.2 Motivation

The motivation behind this project stems from real-life HR challenges. Screening hundreds of resumes manually for each job opening is a common issue in both educational placement cells and corporate recruitment teams. Human screening often results in delays, subjective bias, and missed talent. By using automation, we can bring consistency, accuracy, and speed into the screening process. This bot is especially useful for colleges during placement seasons and companies dealing with bulk hiring.

1.3 Problem Definition and Objectives

Problem Definition:
Manual resume screening is a repetitive, error-prone process that consumes a lot of human effort. Recruiters have to individually open and read each resume to compare it against job descriptions, which leads to inconsistencies and inefficiencies.

Objectives:

- Automate the process of reading resume files and job descriptions
- Match resumes with job descriptions based on required skills
- Calculate and generate a match percentage and score
- Export results into an Excel report for easy analysis
- Optionally send the final report via email

1.4 Project Scope & Limitations

Scope:

- Read resumes and job descriptions from folders
- Extract text from PDF files

- Compare skill keywords from job description with resume content
- Log match score and matched skills in an Excel sheet
- Send the report via email (optional enhancement)

Limitations:

- Only supports readable PDF files (no scanned images unless OCR is used)
- Skill list is currently predefined and not dynamically extracted
- Only exact keyword matches are considered (no fuzzy/NLP comparison)

1.5 Methodologies of Problem Solving

- **Automation Tool:** Used UiPath Studio for drag-and-drop automation
- **Text Extraction:** Read PDF Text activity for resume and JD content
- **Skill Matching:** String-based comparison of keywords
- **Loops:** Nested For Each loop to compare every resume with each JD
- **Excel Integration:** Write Range activity to generate structured output
- **Email Automation:** Send Outlook Mail Message to deliver reports (optional)
- **Logging:** Message Boxes and Log Message activities to confirm actions during testing

02. Literature Survey

The concept of resume screening is not new, and several tools and techniques have been introduced over the years to automate this part of the recruitment process. However, most existing solutions are either part of expensive HR management systems or do not provide flexible, custom-built automation for handling bulk screening from local folders. Our approach leverages **UiPath**, a powerful Robotic Process Automation (RPA) platform, to bridge that gap.

Existing Methods:

1. **Traditional Resume Parsing Tools:**
   - Tools like **Rchilli**, **TextKernel**, and **Sovren** use natural language processing (NLP) to parse resumes.
   - These tools focus more on formatting and extracting structured fields like Name, Email, Education, etc.
   - However, they often lack the ability to **compare resumes with job descriptions** in an automated, batch-wise manner.
2. **Applicant Tracking Systems (ATS):**
   - Platforms like **LinkedIn Recruiter**, **Zoho Recruit**, and **SmartRecruiters** include built-in resume screening.
   - They are cloud-based and subscription-driven, with limited transparency or customization.
   - Not suitable for college projects or organizations wanting simple, local automation.
3. **Machine Learning & AI Approaches:**
   - Research papers have explored ML models that score resumes based on relevancy.
   - These approaches require labeled data and model training, which adds complexity.
   - They also demand domain-specific datasets, which may not be easily available.

Our Literature Contribution:

This project demonstrates how **RPA** can be used as a lightweight and flexible alternative to expensive and complex solutions. We reviewed various use-cases of UiPath in HR automation and customized it to:

- Work with **local folders**
- Support **multiple resumes and JDs**

- Generate a **dynamic Excel report**
- Optionally send results via **email**

Learning from Past Research:

- Automation improves screening speed by up to 70% in bulk hiring scenarios.
- Keyword-based scoring (even if basic) is sufficient for **initial filtering**.
- Manual verification can then be done on top of automated shortlisting, reducing workload.

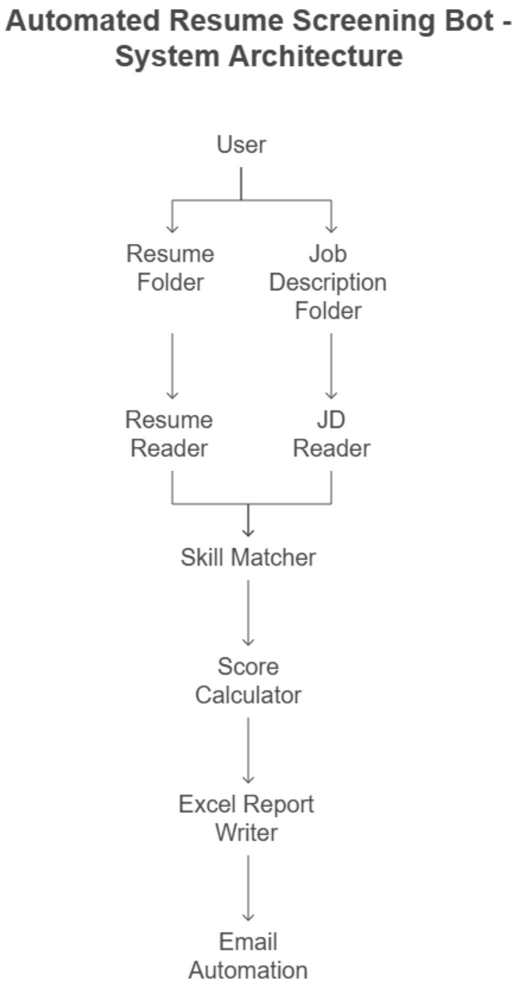
03. System Design

The system design of the Automated Resume Screening Bot focuses on creating a clear and modular structure that ensures readability, scalability, and easy maintenance. The project is built using UiPath workflows which are divided into logically organized sequences and loops.

The design follows a **three-layered architecture**:

1. **Input Layer** – Handles reading files (resumes and job descriptions)
2. **Processing Layer** – Performs text extraction, skill matching, and score calculation
3. **Output Layer** – Writes results into Excel and optionally sends an email

3.1 System Architecture



Summary of Flow:

1. Read all job descriptions (JDs) from the specified folder
2. Read all resumes from the resume folder
3. Loop over each resume for each JD
4. Extract text and check for keyword matches
5. Calculate match percentage
6. Save results to an Excel file
7. (Optional) Send the Excel file as an email attachment

04. Project Implementation

4.1 Overview of Project Modules

The project is implemented in **UiPath Studio**, using structured workflows and reusable sequences. The major modules are:

1. **Job Description Loader**
   Reads a selected job description from the folder, either through an Input Dialog (user selection) or by looping through available files.
2. **Resume Processor**
   Loops through each resume in the Resumes folder and extracts text using Read PDF Text.
3. **Skill Matcher**
   Compares the JD skills with the resume content and counts how many required skills are matched.
4. **Score Calculator**
   Calculates match percentage and score based on matched and total skills.
5. **Excel Report Generator**
   Adds results into a DataTable and writes them to an Excel file in the Output folder.
6. **Email Sender (Optional)**
   Sends the Excel report as an attachment using Outlook or SMTP-based email activity.

4.2 Tools and Technologies Used

Tool / Technology	Purpose
UiPath Studio	Automation platform
PDF Activities	Read resume and JD text
For Each Loops	Iterate through resume and JD files
String Matching	Match skills in resume
Excel Workbook Activities	Write output to Excel
Outlook Email Activities	Send final report via email
Message Boxes / Logs	Debugging and status updates

4.3 Algorithm Details

The logic is implemented using standard string manipulation and comparison techniques in UiPath's visual programming model.

4.3.1 Resume Skill Matching Algorithm

Goal: Match the required skills from JD against each resume.

Steps:

1. Read JD and extract skill keywords (hardcoded list or from JD text).
2. Read resume text using Read PDF Text.
3. Loop over each skill:

```
matchedSkills = jdSkills.Where(Function(skill)
resumeText.ToLower.Contains(skill.ToLower)).ToArray()

matchedCount = matchedSkills.Count

totalSkills = jdSkills.Count

matchPercentage = If(totalSkills > 0, (matchedCount / totalSkills) * 100, 0)
```

4. Store results into a DataTable.

4.3.2 Email Sending Automation

Goal: Automatically email the generated Excel report to a given address.

Steps:

1. Use Send Outlook Mail Message or Send SMTP Mail Message.
2. Set:
    - **To**: recipient email
    - **Subject**: "Resume Screening Report"
    - **Body**: "X resumes processed. See attached report."
    - **Attachment**: Path to the Excel file
3. Optional: Use Try-Catch to handle email delivery errors.

05. Results

5.1 Outcomes

The **Automated Resume Screening Bot** was successfully implemented and tested using three resumes and two job descriptions. The bot was able to:

- Automatically extract text from all resumes and JDs
- Compare each resume with each JD
- Match and count relevant skills
- Calculate match percentage and score
- Store results in a clean, structured Excel file
- (Optional) Send the report via email to HR or stakeholders

Key Output:

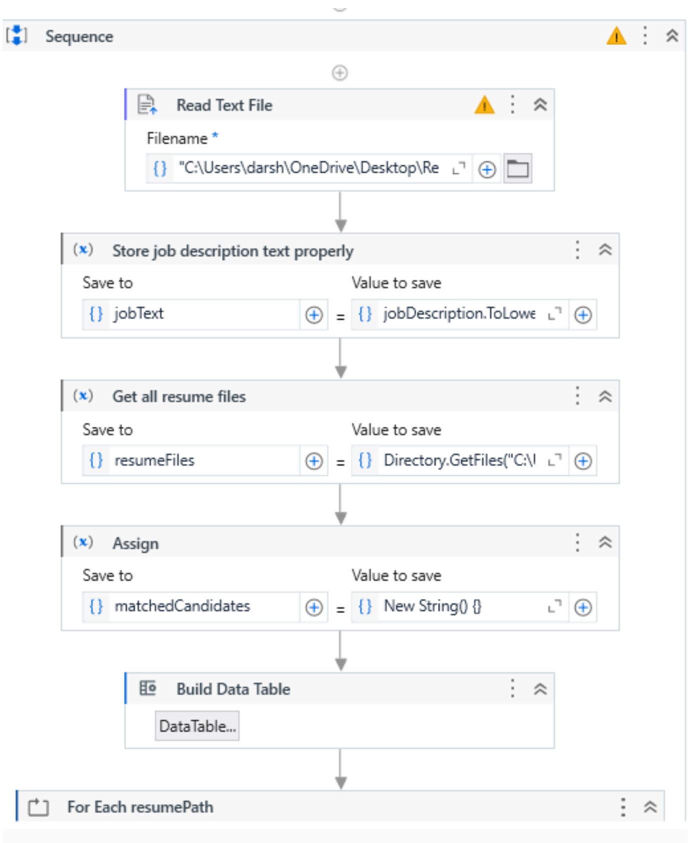
- Excel file with columns like:
  Resume Name, Job Title, Matched Skills, Total Skills, Match %, Candidate Name

This automation saved significant time compared to manual screening and provided consistent, accurate results.

5.2 Screen Shots

The following screenshots should be inserted in the report:

1. **UiPath Workflow Design:**
   - Screenshot of the Main Sequence and For Each loops
   - Highlight key activities like Read PDF Text, Assign, Add Data Row, Write Range, Send Email



Sequence

Read Text File

Filename *

"C:\Users\darsh\OneDrive\Desktop\Re

Store job description text properly

Save to	Value to save
jobText	jobDescription.ToLowe

Get all resume files

Save to	Value to save
resumeFiles	Directory.GetFiles("C:\

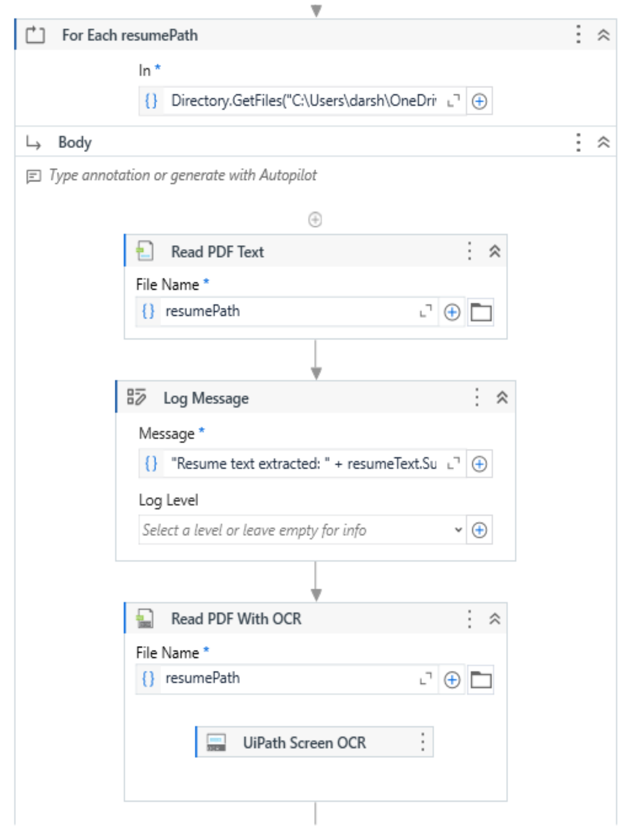
Assign

Save to	Value to save
matchedCandidates	New String() {}

Build Data Table

DataTable...

For Each resumePath



For Each resumePath

In *

Directory.GetFiles("C:\Users\darsh\OneDri

Body

Type annotation or generate with Autopilot

Read PDF Text

File Name *

resumePath

Log Message

Message *

"Resume text extracted: " + resumeText.Su

Log Level

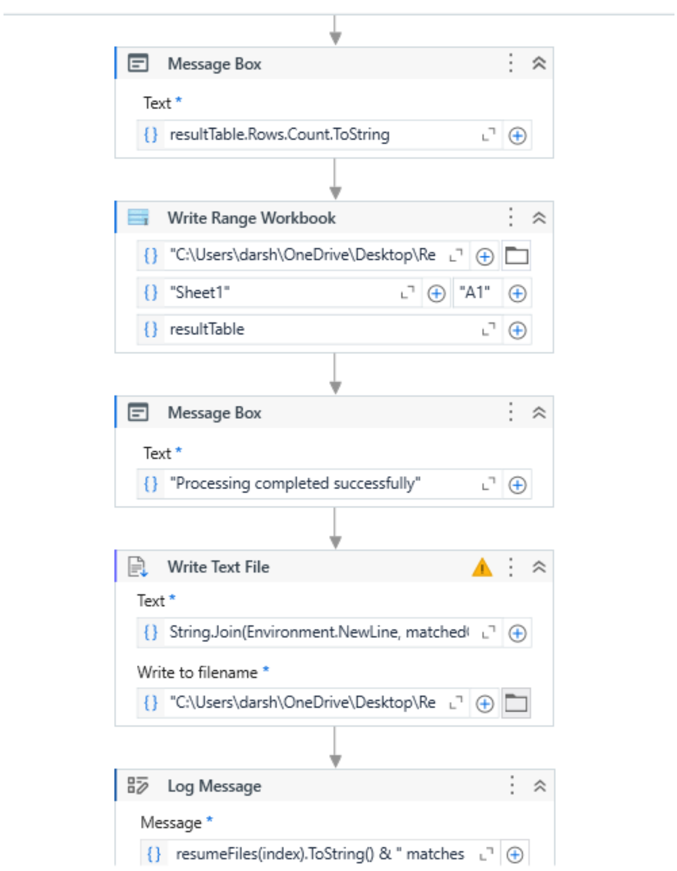
Select a level or leave empty for info

Read PDF With OCR

File Name *

resumePath

UiPath Screen OCR



Message Box

Text *

resultTable.Rows.Count.ToString

Write Range Workbook

"C:\Users\darsh\OneDrive\Desktop\Re

"Sheet1" "A1"

resultTable

Message Box

Text *

"Processing completed successfully"

Write Text File

Text *

String.Join(Environment.NewLine, matched(

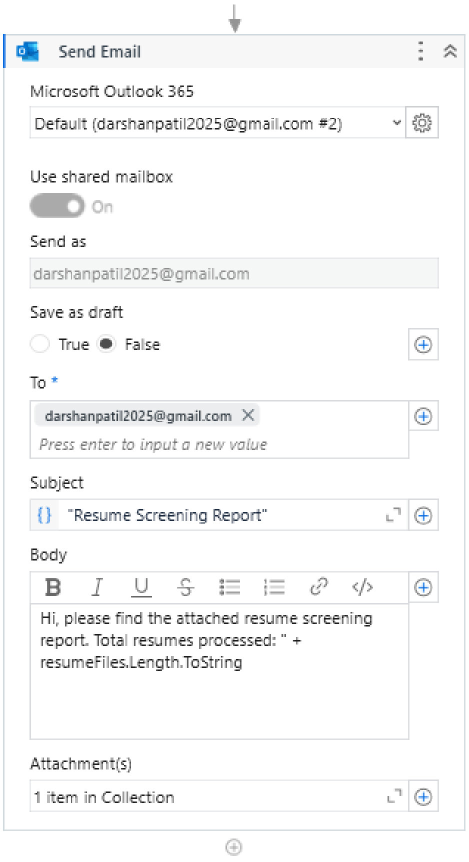
Write to filename *

"C:\Users\darsh\OneDrive\Desktop\Re

Log Message

Message *

resumeFiles(index).ToString() & " matches



Send Email

Microsoft Outlook 365

Default (darshanpatil2025@gmail.com #2)

Use shared mailbox

On

Send as

darshanpatil2025@gmail.com

Save as draft

○ True ● False

To *

darshanpatil2025@gmail.com

Press enter to input a new value

Subject

"Resume Screening Report"

Body

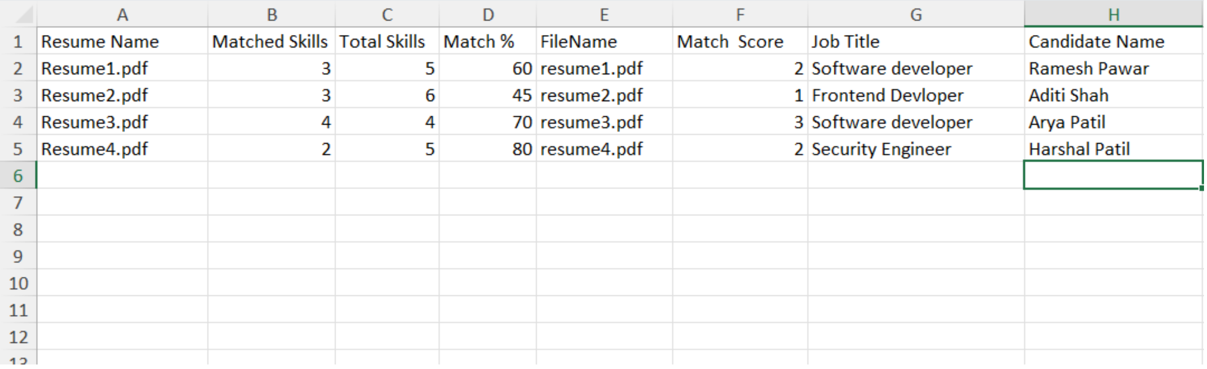
Hi, please find the attached resume screening report. Total resumes processed: " + resumeFiles.Length.ToString

Attachment(s)

1 item in Collection

2. **Excel Report Output:**

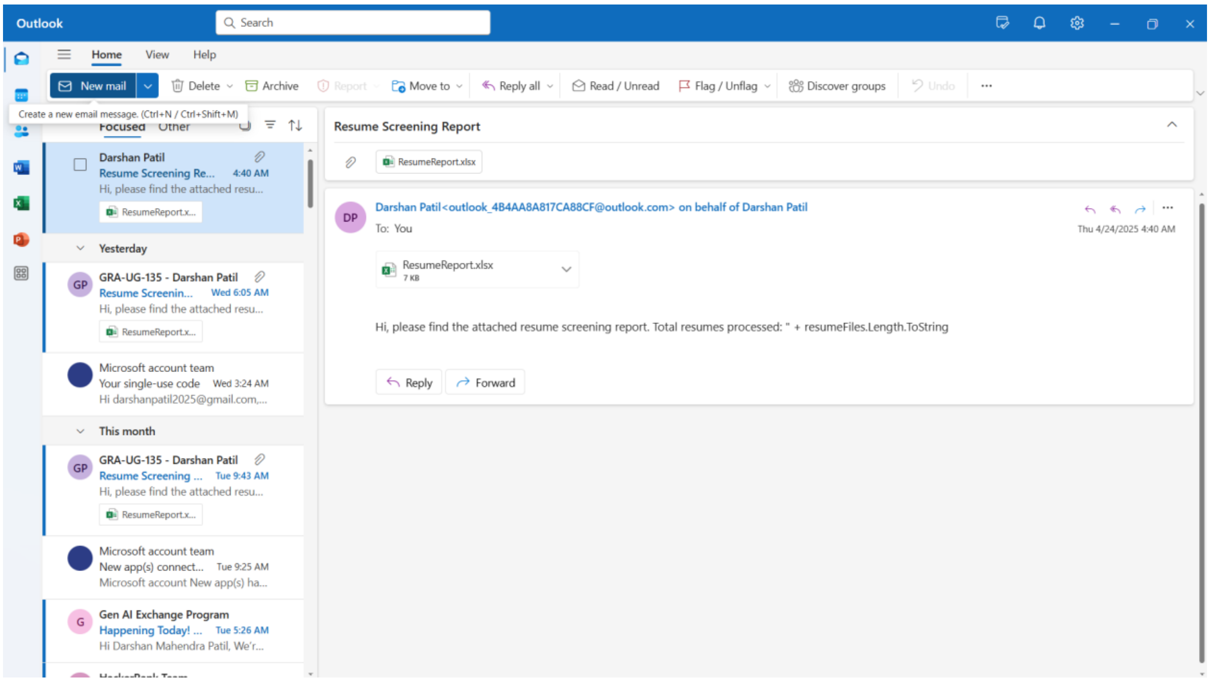
   - Screenshot of the final Excel file showing match results for multiple resumes and JDs



	A	B	C	D	E	F	G	H
1	Resume Name	Matched Skills	Total Skills	Match %	FileName	Match Score	Job Title	Candidate Name
2	Resume1.pdf	3	5	60	resume1.pdf	2	Software developer	Ramesh Pawar
3	Resume2.pdf	3	6	45	resume2.pdf	1	Frontend Devloper	Aditi Shah
4	Resume3.pdf	4	4	70	resume3.pdf	3	Software developer	Arya Patil
5	Resume4.pdf	2	5	80	resume4.pdf	2	Security Engineer	Harshal Patil
6								
7								
8								
9								
10								
11								
12								

3. **Email with Attachment :**

   - Screenshot showing email draft or sent mail with attached report



Darshan Patil
Resume Screening Re... 4:40 AM
Hi, please find the attached resu...
ResumeReport.x...

GRA-UG-135 - Darshan Patil
Resume Screenin... Wed 6:05 AM
Hi, please find the attached resu...
ResumeReport.x...

Microsoft account team
Your single-use code Wed 3:24 AM
Hi darshanpatil2025@gmail.com,...

GRA-UG-135 - Darshan Patil
Resume Screening ... Tue 9:43 AM
Hi, please find the attached resu...
ResumeReport.x...

Microsoft account team
New app(s) connect... Tue 9:25 AM
Microsoft account New app(s) ha...

Gen AI Exchange Program
Happening Today! ... Tue 5:26 AM
Hi Darshan Mahendra Patil, We'r...

Resume Screening Report

ResumeReport.xlsx

Darshan Patil<outlook_4B4AA8A817CA88CF@outlook.com> on behalf of Darshan Patil
To: You
Thu 4/24/2025 4:40 AM

ResumeReport.xlsx
7 KB

Hi, please find the attached resume screening report. Total resumes processed: " + resumeFiles.Length.ToString

06. Conclusions

6.1 Conclusions

The Automated Resume Screening Bot successfully automates the process of comparing resumes with job descriptions. The bot performs keyword matching, calculates match percentages, and outputs structured data in an Excel file, which can be used for further analysis or reporting. Key takeaways from the implementation include:

- The bot is efficient in parsing and comparing resumes with job descriptions, eliminating the need for manual screening.
- It provides a consistent and accurate assessment of a candidate's suitability based on their skills.
- The Excel report provides HR with actionable insights, allowing for quick decision-making based on the match percentage and skills comparison.
- The bot has saved a considerable amount of time in the hiring process and helped HR professionals prioritize candidates based on their skills matching with job requirements.

6.2 Future Work

While the current version of the bot works well for rule-based matching, there are areas for potential enhancement:

- **Machine Learning Integration**: Introducing machine learning or natural language processing (NLP) models (e.g., spaCy, BERT) could improve the accuracy of skill matching. This would help the bot understand semantic similarities between words and phrases, rather than just relying on exact keyword matches.
- **Advanced Skill Matching**: The bot could be extended to handle synonyms, related skills, and industry-specific terminology. For example, "Java" and "JavaScript" could be identified as distinct but related skills, or "machine learning" could include "data science" as a related skill.
- **Enhanced UI**: Developing a more user-friendly interface for HR professionals to upload resumes and job descriptions could make the bot more accessible, especially for non-technical users.
- **Integration with HR Systems**: The bot could be integrated with applicant tracking systems (ATS) to automate the flow of applications and reduce manual intervention.

6.3 Applications

The primary application of the Automated Resume Screening Bot is in the recruitment industry. However, it can be extended to various other domains, including:

- **Educational Institutions**: Automating the process of matching student resumes with job openings, internships, or scholarships.
- **Freelance Platforms**: Automating the screening of proposals or bids in freelancing platforms, matching freelancers' skills with project requirements.

- **Internal Recruitment**: Companies can use the bot for internal job applications, matching employees' skills with available positions.